เรื่องน่าอ่าน
สมชาย นำประเสริฐชัย

สารพัน...UNIX
ตอน ไฟล์ผู้ไม่โดดเดี่ยว

ศึกษาระบบโครงสร้างไฟล์ของยูนิกซ์ และกรรมวิธีในการจัดการกับข้อมูลของระบบ

ยูนิกซ์เป็นโอเอสที่สนับสนุนให้มีการใช้งานได้หลาย ๆ คนในเวลาเดียวกัน หรือเรียกว่ายูนิกซ์เป็นโอเอสแบบมัลติยูสเซอร์ (Multi-user) และยังสามารถให้มีการใช้งานข้อมูลต่าง ๆ ร่วมกันได้ จากเหตุข้างต้นนี้เองส่งผลให้จำนวนและขนาดของข้อมูลที่จะมาสนับสนุนการใช้งานมีเป็นจำนวนมาก ดังนั้นยูนิกซ์จึงจำเป็นต้องมีการจัดโครงสร้างและกรรมวิธีต่าง ๆ ในการจัดการกับบรรดาข้อมูล เพื่อให้เกิดความคล่องตัวและมีประสิทธิภาพสูงที่สุดเท่าที่จะเป็นไปได้

ระบบไฟล์ (File systems)

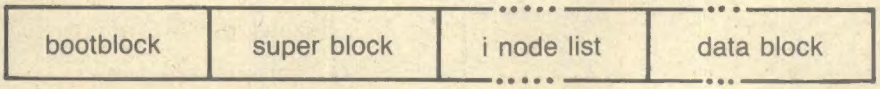
ยูนิกซ์จะมีการแบ่งเนื้อที่ของฮาร์ดดิสค์ออกเป็นส่วน ๆ แต่ละส่วนจะถูกจัดว่าเป็นระบบไฟล์ (file systems) โดยระบบไฟล์นี้เองจะเป็นที่เก็บของบรรดาไฟล์ทั้งหมด ซึ่งการกำหนดจำนวนระบบไฟล์นั้น อาจจะมีเพียง 1 ระบบไฟล์หรือมากกว่าก็ได้ ขึ้นอยู่กับความต้องการและขั้นตอนการติดตั้ง และในปัจจุบันนั้นไฟล์ซิสเต็มอาจจะอยู่ในระบบเครือข่ายคอมพิวเตอร์แทนก็ได้

การจัดเก็บไฟล์ในยูนิกซ์จะเก็บในรูปของบล็อก (Block) โดยจะมีขนาดเป็นจำนวนเท่าของ 512 ไบต์ ขึ้นอยู่กับการกำหนดขนาดของระบบนั้น ๆ ซึ่งการกำหนดขนาดเล็กหรือใหญ่ก็มีข้อดีข้อเสียแตกต่างกันไป หากบล็อกมีขนาดใหญ่จะทำให้การแลกเปลี่ยนข้อมูลระหว่างดิสค์กับหน่วยความจำทำได้รวดเร็ว แต่อาจจะทำให้เกิดการสูญเสียเนื้อที่ (fragmentation) ไปได้โดยง่าย การจัดเก็บโดยปกติจะเก็บเป็นบล็อกที่ติดต่อกัน แต่ถ้ามีไฟล์อื่นมาแทรกก็สามารถกระโดดข้ามบล็อกนั้นไปได้ เนื่องจากมีตารางบอกว่าไฟล์นั้นเก็บข้อมูลอยู่ที่บล็อกใดบ้าง โดยเก็บไว้ในส่วนของ i node ซึ่งจะกล่าวในลำดับต่อไป

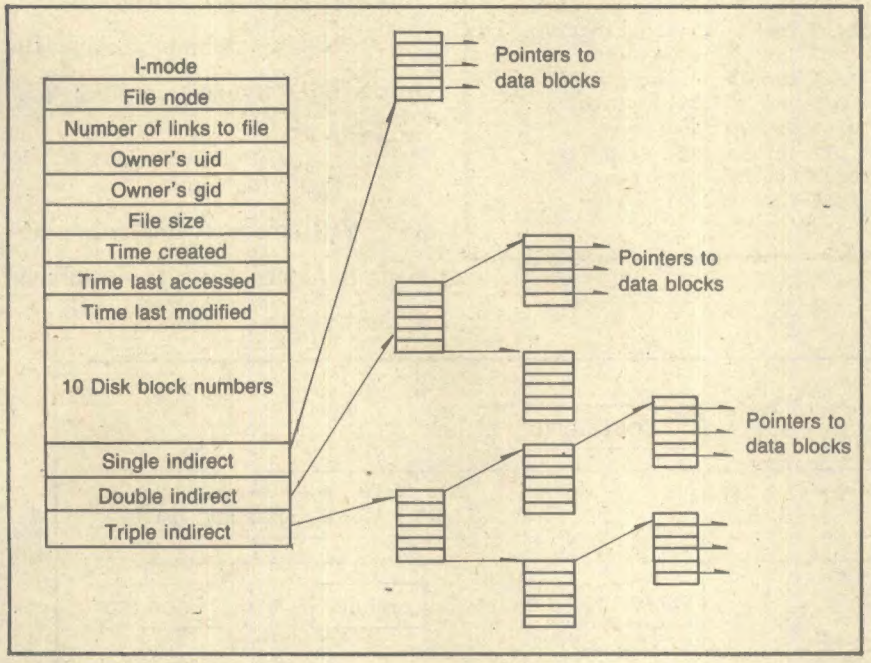
ส่วนประกอบของระบบไฟล์

ส่วนประกอบของระบบไฟล์มีอยู่ 4 ส่วน ดังรูปที่ 1 คือ

1. boot block เป็นส่วนแรกของ



รูปที่ 1 ส่วนประกอบของระบบไฟล์



รูปที่ 2 แสดงโครงสร้างของ i node

ไฟล์ซิสเต็ม โดยเป็นส่วนที่ทำหน้าที่ในการเริ่มต้นการทำงานของระบบ แม้ว่าเพียง 1 บูตบล็อก ก็พอสำหรับการเริ่มต้นทำงานของระบบ แต่จะพบว่าทุกไฟล์ซิสเต็มจะมีส่วนของบูตบล็อกรวมอยู่ด้วย

2. Super block เป็นส่วนที่บอกถึงสถานะของระบบไฟล์ ทั้งขนาด, จำนวนไฟล์ที่สามารถเก็บได้ และข้อมูลอื่น ๆ

3. i node list เก็บบรรดา i node ต่าง ๆ ซึ่ง i node จะเป็นส่วนที่สำคัญมาก โดยจะกล่าวในลำดับต่อไป

4. data block เป็นส่วนที่เก็บเนื้อแท้ของไฟล์ประเภทต่าง ๆ

I node คืออะไร

เคอร์เนลจะไม่สามารถติดต่อหรือรู้จักไฟล์โดยตรง แต่เคอร์เนลจะรับรู้สิ่งต่าง ๆ เกี่ยวกับไฟล์ที่มีอยู่ได้จากโครงสร้างข้อมูลที่เรียกว่า i node โดย i node จะประกอบด้วยข้อมูลต่าง ๆ เกี่ยวกับไฟล์เป็นจำนวนมาก ดังรูปที่ 2 เช่น

- การแสดงสิทธิต่าง ๆ ของไฟล์
- การบอกถึงชนิดของไฟล์
- แสดงถึงเจ้าของ และกลุ่มของเจ้าของไฟล์
- วัน เวลา ที่สร้างไฟล์, เปลี่ยนแปลงไฟล์ หรือแอ็กเซสไฟล์
- จำนวน link ที่ไฟล์เชื่อมต่อด้วย
- เก็บตำแหน่งที่อยู่ของข้อมูลในไฟล์นั้น ๆ ซึ่งการอ้างถึงตำแหน่งของข้อมูลนั้นสามารถอ้างโดยตรง (direct pointer) หรืออ้างโดยการผ่านตัวอื่นก่อน (Indirect pointer) โดยในส่วนนี้ยังแบ่งออกเป็น single indirect, double indirect และ triple indirect

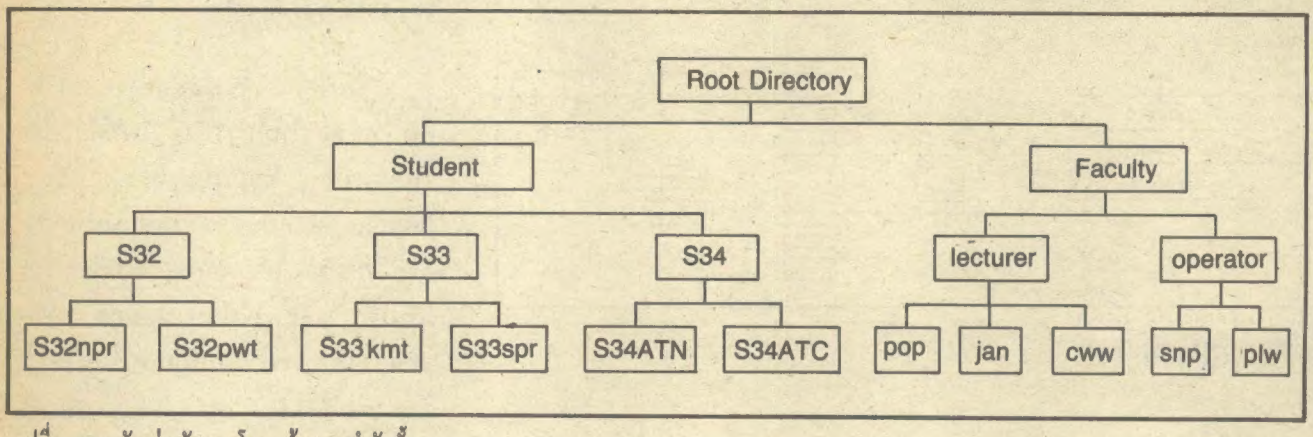
i node นั้นจะถูกสร้างขึ้นมาโดยอัตโนมัติตอนสร้างระบบไฟล์ และถ้าหาก i node นั้นเต็มจะไม่สามารถสร้างไฟล์ขึ้นมาได้เลย ถึงแม้ว่าจะยังมีเนื้อที่ว่างอยู่

การตั้งชื่อไฟล์

ชื่อไฟล์ของยูนิกซ์นั้นต่างจากดอส คือ ดอสจะให้มีการตั้งชื่อไฟล์ด้วยชื่อไฟล์ขนาด 8 ตัวอักษร และสกุลของไฟล์อีก 3 ตัวอักษร ส่วนยูนิกซ์นั้นการตั้งชื่อสามารถใช้ตัวอักษร ตัวเลข ขีดเส้นใต้ และยังให้ความแตกต่างกันระหว่างอักษรตัวเล็กกับอักษรตัวใหญ่ด้วย เช่น FILE 1, file 1, File 1 ยูนิกซ์จะถือว่าทั้ง 3 ไฟล์นั้นมีชื่อแตกต่างกัน ส่วนความยาวของชื่อไฟล์นั้นไม่จำกัด แต่จะถูกกำหนดโดยเคอร์เนลว่าจะต้องไม่เกิน 255 ตัวอักษร และความยาวของทั้ง path นั้นห้ามเกิน 1023 ตัวอักษร ส่วนชื่อที่จะตั้งนั้นก็เหมือนกับการตั้งชื่อทั่วไป คือ ให้มีความหมายเป็นนัยบอกถึงหน้าที่ หรือลักษณะของไฟล์ เพราะจะทำให้เข้าใจและใช้งานได้สะดวกยิ่งขึ้น การตั้งชื่อนั้นควรหลีกเลี่ยงการใช้เครื่องหมายพิเศษต่าง ๆ เช่น ^ " ' , - ? [ ] ( ) ~ ! $ { } < > # @ & | และยิ่งกว่านั้นถ้าไฟล์ใดที่มีชื่อขึ้นต้นด้วย (dot) จะกลายเป็นไฟล์ที่ถูกซ่อนไว้ (hidden file) ซึ่งจะไม่แสดงรายชื่อไฟล์ออกมาให้เห็น หากต้องการเรียกไฟล์ที่ถูกซ่อนขึ้นมาดู

```
$ ls -al
total 130
drwxr-xr-x   6 snp      operator     512 Jun 16 13:07 .
drwxr-x---   4 root     operator     512 Jul 20 1991  ..
-rw-r--r--   1 snp      operator     104 Jul 19 1991  .Xdefaults
-rwxr-----   1 snp      operator     547 Mar 13 14:15 .profile
-rw-------   1 snp      operator    3228 Jun 16 13:25 .sh_history
-rwxr-xr-x   1 snp      system      2413 Feb 13 10:43 .xinitrc
-rw-r--r--   1 snp      operator       0 Jun 16 13:04 micro
-rw-r--r--   1 snp      operator     311 Jun 16 13:06 micro4
-rw-r--r--   1 snp      operator    1015 Jun 16 13:24 micro5
-rwx------   1 snp      operator    8242 Sep 30 1991  snp-
drwx------   2 snp      operator     512 Jul 19 1991  supplies
-rwxr--r--   1 snp      operator   10371 Feb 06 10:43 test
drwx------   2 snp      operator     512 Jul 19 1991  waste
```

รูปที่ 3



รูปที่ 4 แสดงตัวอย่างลักษณะโครงสร้างแบบลำดับชั้น

ต้องเพิ่มพารามิเตอร์พิเศษ (-a) ให้กับคำสั่งในการขอดูรายชื่อไฟล์ ดังรูปที่ 3

ลักษณะโครงสร้างของไฟล์ในยูนิกซ์

ระบบโครงสร้างไฟล์ของยูนิกซ์มีการจัดเป็นแบบลำดับชั้น (herachical) โดยมีลักษณะคล้ายกับต้นไม้กลับหัว โดยแสดงดังรูปที่ 4

โดยจุดเริ่มต้นหรือชั้นแรกจะเรียกว่า เป็นราก (root) ซึ่งจะเขียนแทนด้วยเครื่องหมาย / ไฟล์แต่ละไฟล์นั้น อาจจะเป็นข้อมูลที่สร้างขึ้นหรือโปรแกรมก็ได้ แม้แต่จะเก็บไฟล์เองก็ยังได้ ซึ่งไฟล์ลักษณะนี้จะจัดว่าเป็นไฟล์ประเภทไดเรคทอรี รูปแบบการจัดเก็บไฟล์แบบลำดับชั้นนี้จะทำให้การจัดเป็นไฟล์เป็นระบบ ทำให้ง่ายต่อการใช้งานและดูแลแต่ปัญหาที่เกิดขึ้นก็คือการสูญเสียเวลาที่ใช้ในการค้นหาไฟล์ที่ต้องการ เนื่องจากไฟล์มีจำนวนมาก

ประเภทของไฟล์

ไฟล์ในระบบยูนิกซ์ ได้มีการแบ่งประเภทออกไปหลายรูปแบบ และรูปแบบนั้นยังขึ้นอยู่กับกลุ่มของผู้สร้างยูนิกซ์ด้วย

- Regular file (ไฟล์ทั่วไป)
- Directories (ไดเรคทอรี)
- Character device file และ Block device file หรือเรียกรวม ๆ ว่า Block and Character device file
- UNIX - domain sockets ในยูนิกซ์ BSD หรือ Name pipes ของยูนิกซ์ ATT
- Symbolic - link

ซึ่งประเภทของไฟล์นี้จะแสดงโดยคำสั่ง ls ดังรูปที่ 5 และสัญลักษณ์ของประเภทของไฟล์แสดงตารางรูปที่ 6

ไฟล์ทั่วไป (Regular file)

เป็นไฟล์ธรรมดาที่อาจสร้างขึ้นเองจากเอดิเตอร์ หรืออาจจะทำการก๊อบปี้มาจากไฟล์อื่น หรืออาจจะเป็นโปรแกรมใช้งานต่าง ๆ ก็ได้ โดยลักษณะการรายงานรายละเอียดต่าง ๆ ของไฟล์ เช่น สิทธิในไฟล์ เจ้าของไฟล์ กลุ่มเจ้าของไฟล์ ขนาดหรือวันเวลาที่ทำการสร้าง เปลี่ยนแปลงแก้ไขไฟล์ ซึ่งจะได้ข้อมูลต่าง ๆ นี้มาจาก i node นั่นเอง ดังรูปที่ 7

ไดเรคทอรี (Directories)

เป็นไฟล์อื่นชนิดหนึ่งซึ่งใช้เก็บบรรดาไฟล์หรือไดเรคทอรีด้วยกันเอง

```
total 39
drwxrwxr-x   2 root     system      2560 Jun 16 07:50 .
drwxr-xr-x  19 bin      bin         1024 Jun 15 19:33 ..
srwxrwx---   1 root     system         0 Jun 16 07:50 SRC
cr--r----T   1 root     system      8,  0 Jun 26 1991  audit
crw-r----T   1 root     system      3,  8 Jun 26 1991  bus0
br--r--r--   1 root     system     21,  0 Jun 26 1991  cd0
-r--r--r--   1 root     system         0 Jun 26 1991  cfglock
brw-rw-rw-   1 root     system     14,  0 Apr 09 16:21 fd0
brw-rw-rw-   1 root     system     14,  1 Jun 26 1991  fd0h
brw-rw-rw-   1 root     system     14,  2 Jun 26 1991  fd01
lrwxrwxrwx   1 root     system        10 Jun 26 1991  ptyp0 -> /dev/ptc/0
lrwxrwxrwx   1 root     system        10 Jun 26 1991  ptyp1 -> /dev/ptc/1
lrwxrwxrwx   1 root     system        10 Jun 26 1991  ptyp4 -> /dev/ptc/4
```

รูปที่ 5

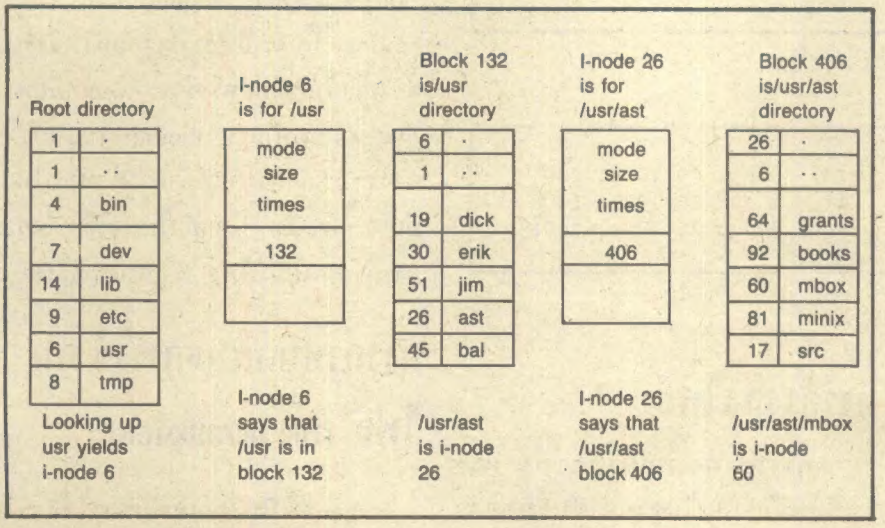
File–type encoding used by ls

File type	Symbol	Created by	Removed by
Regular file	"–"	editors,cp,etc.	rm
Directory	"d"	mkdir	rmdir,rm –r
Character device file	"c"	mknod	rm
Block device file	"b"	mknod	rm
UNIX domain socket (BSD)	"s"	socket	rm
Named pipe (ATT)	"p"	mknod	rm
Symbolic link (BSD)	"l"	ln –s	rm

รูปที่ 6

```
$ ls -l
total 111
-rw-------   1 snp      operator     2916 Jun 11 13:20 mbox
-rw-r--r--   1 snp      operator        0 Jun 16 13:04 micro
-rw-r--r--   1 snp      operator      311 Jun 16 13:06 micro4
-rw-r--r--   1 snp      operator        0 Jun 16 13:07 micro5
-rwx------   1 snp      operator     8242 Sep 30 1991  snp~
drwx------   2 snp      operator      512 Jul 19 1991  supplies
-rwx------   1 snp      operator    10371 Feb 06 10:43 test
drwx------   2 snp      operator      512 Jul 19 1991  waste
-rwx------   1 snp      operator    10371 Jul 06 10:43 test
```

รูปที่ 7



รูปที่ 8 ขั้นตอนการอ้างถึงไฟล์ของไดเรคทอรี

ก็ได้ รูปแบบกรรมวิธีการอ้างถึงไฟล์ตามไดเรคทอรีนั้นกระทำโดยการอ่านค่าตำแหน่งจาก i node ว่าไฟล์ที่ต้องการนั้นอยู่ในบล็อกใด ดังตัวอย่างในรูปที่ 8 นอกจากนี้ยังมีการแบ่งประเภทไดเรคทอรีให้เป็นไปตามหน้าที่ของกลุ่มไฟล์ที่อยู่ในไดเรคทอรีนั้น ๆ จนยูนิกซ์จากหลาย ๆ ค่ายแทบจะมีการจัดเก็บไฟล์ต่าง ๆ ในไดเรคทอรีที่มีชื่อเหมือน ๆ กัน หรือคล้าย ๆ กัน ดังแสดงในตาราง รูปที่ 9

ไฟล์พิเศษ (Block & Character device file)

ยูนิกซ์มีไฟล์ชนิดพิเศษอยู่ 2 ชนิด คือ block และ character device file โดยไฟล์เหล่านี้จะมีดีไวซ์ไดรเวอร์เป็นส่วนจัดการไฟล์ โดยส่วนใหญ่ไฟล์พิเศษเหล่านี้จะอยู่ในไดเรคทอรี -dev คือจะเป็นบรรดาฮาร์ดแวร์ หรืออุปกรณ์ต่าง ๆ ของคอมพิวเตอร์ เช่น ฮาร์ดดิสก์, ดิสก์ไดรฟ์, จอภาพ ฯลฯ

การแสดงรายละเอียดของไฟล์พิเศษนี้ต่างไปจากไฟล์ทั่วไปและไดเรคทอรี คือจะไม่บอกขนาดของไฟล์ แต่จะแสดงตัวเลข 2 แถวแทน โดยตัวเลขแต่ละตัวมีความหมายต่างกันไป ตัวเลขตัวแรกเป็นการบอกเคอร์เนลว่าไฟล์นั้นอ้างถึงดีไวซ์ไดรเวอร์ไหนซึ่งตัวเลขนี้จะเรียกว่า เมเจอร์ (major) ส่วนตัวเลขแถวที่สองจะเป็นส่วนที่บอกกับดีไวซ์ไดรเวอร์ว่าไฟล์พิเศษนั้นอ้างถึงอุปกรณ์ (Physical unit) ตัวเลขหลักนี้เรียกว่าเป็น ไมเนอร์ (minor)

จากตัวอย่างในรูปที่ 10 จะเห็นได้ว่า สามารถใช้ดีไวซ์ไดรเวอร์ตัวหนึ่งอ้างถึงอุปกรณ์กายภาพได้หลายค่า เช่น เวลาอ้างถึงฟลอปปี้ดิสก์ชนิดความหนาแน่นสูงจะใช้ไฟล์หนึ่ง และเมื่ออ้างถึง

Standard directories and their contents

Pathname	Contents
/	The root directory, contains the UNIX binary image
/bin	Commands needed for minimal system operability
/5bin	System V compatibility commands for BSD-based systems
/dev	Device entries for terminals, disks, modems, etc.
/etc	Critical system configuration files and maintenance commands
/lib	The C compiler and preprocessor
/tmp	Temporary files – usually a small amount of space
/sys	Kernel building work area, kernel configuration files (BSD)
/stand	Standalone utilities (disk formatters, etc.)
/usr/adm	Accounting files, records of resource usage
/usr/bin	Executable files
/usr/dict	The spelling dictionary (BSD)
/usr/doc	Larger UNIX-related documents, no manual pages
/usr/games	Games and diversions
/usr/include	All UNIX-defined C header files
/usr/etc	Where Sun puts things that everyone else puts in/etc
/usr/lib	Support files for standard UNIX programs
/usr/man	Online manual pages
/usr/old	Obsolete software that will soon be unsupported (BSD)
/usr/new	New software that will soon be supported (BSD)
/usr/preserve	Backup copies of files created by the editors vi and ex
/usr/spool	Spooling directories for printers, uucp, mail, etc.
/usr/tmp	More temporary space
/usr/src	Source code for UNIX standard commands, if you have it
/usr/src/uts	Kernel building work area, kernel configuration files (ATT)
/usr/ucb	Berkeley utilities and programs (BSD)
/usr/Ibin	Local binaries (ATT)
/usr/local	Local software (BSD)
/usr/local/adm	Accounting log files
/usr/local/bin	Executables
/usr/local/etc	System maintenance commands
/usr/local/lib	Support files
/usr/local/src	Source for/usr/local/bin,/usr/local/lib, and/usr/local/etc
/usr/src/local	Alternative location for/usr/local/src

รูปที่ 9 แสดงลักษณะของไฟล์ที่อยู่ในไดเรคทอรี

```
brw-rw-rw-   1 root      system      14,  0 Apr 09 16:21 fd0
brw-rw-rw-   1 root      system      14,  1 Jun 26 1991  fd0h
brw-rw-rw-   1 root      system      14,  2 Jun 26 1991  fd01
```

รูปที่ 10

ฟลอปปี้ดิสค์ชนิดความหนาแน่นต่ำก็ใช้อีกไฟล์หนึ่ง โดยทั้งสองไฟล์จะต่างกันเพียงส่วนของไมเนอร์ หรือต่างกันเพียงการอ้างถึงอุปกรณ์ว่าเป็นประเภทไหน

การอ้างถึงไฟล์

การอ้างถึงไฟล์หรือการเข้าหาไฟล์นั้นต้องเป็นไปตามลำดับขั้นของโครงสร้างไดเรคทอรี ซึ่งการอ้างถึงไฟล์นั้นสามารถทำได้ 2 วิธี คือ

1. การอ้างถึงแบบสัมพัทธ์ (Relative pathname) โดยการอ้างจากตำแหน่งที่อยู่ขณะนั้น โดยมีสัญลักษณ์พิเศษเข้ามาช่วยในการอ้างอิง คือ

. แสดงถึง ไดเรคทอรีปัจจุบันที่อยู่

. . แสดงถึง ไดเรคทอรีที่อยู่เหนือถัดขึ้นไป 1 ชั้น

2. การอ้างถึงแบบสัมบูรณ์ (Absolute pathname) เป็นการอ้างถึงไฟล์โดยการเริ่มต้นจากตำแหน่งแรก คือ อ้างจากราก (/) เลยทีเดียว แสดงดังรูปที่ 11 และ 12

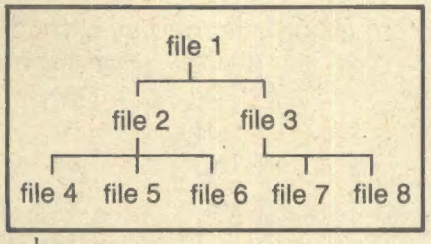
การ mount และ umount ระบบไฟล์

การ mount จะเป็นการนำไฟล์เข้าไปเชื่อมต่อกับไฟล์ทรีที่มีอยู่เดิมแล้ว โดยจะนำไฟล์ที่ต้องการเชื่อมต่อมาทับตรงจุดที่ต้องการเชื่อม ซึ่งจุดนี้จะเรียกว่า (mount point) หลังจากทำการเชื่อมต่อแล้ว จะไม่สามารถมองเห็นไฟล์เดิมที่ถูกทับได้ ดังรูปที่ 13

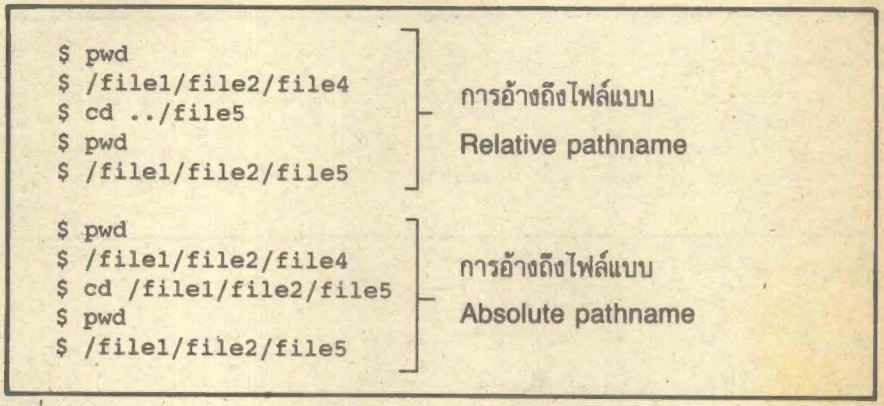
ส่วนคำสั่ง umount หรือ Unmount นั้นจะเป็นการยกเลิกการเชื่อมต่อไฟล์เข้ากับไฟล์ทรี ซึ่งจะได้ผลเหมือนกับตอนก่อนทำการ mount ในยูนิกซ์บางระบบยังสามารถทำการกำหนดให้มีการ mount โดยอัตโนมัติได้ โดยการกำหนดไว้ในไฟล์ /etc/filesystems

การกำหนดสิทธิการใช้งานไฟล์ (file permission)

ยูนิกซ์เป็นโอเอสที่มีการใช้งานไฟล์ต่าง ๆ ร่วมกันในกลุ่มของผู้ใช้งาน



รูปที่ 11



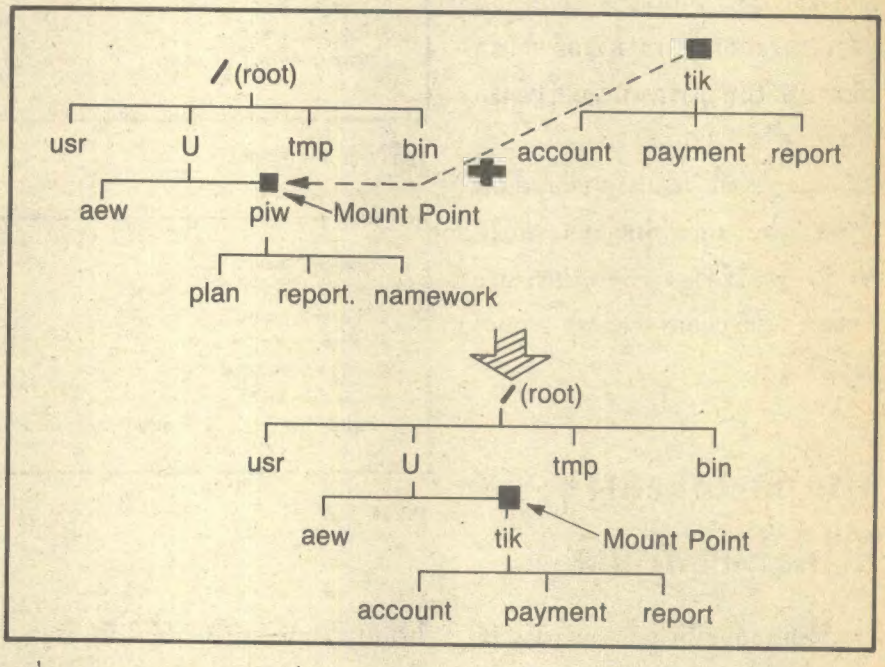
รูปที่ 12

ในระบบ ซึ่งจะก่อให้เกิดความวุ่นวาย หากให้สิทธิแก่ผู้ใช้ทั้งหมดในการจัดการหรือกระทำสิ่งใด ๆ ก็ได้กับบรรดาไฟล์ต่าง ๆ ที่ใช้ร่วมกันอยู่ ดังนั้นยูนิกซ์จึงได้มีการจำกัดสิทธิต่าง ๆ ของไฟล์กับบรรดาผู้ใช้งาน

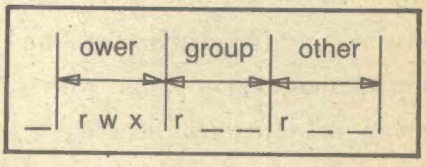
ยูนิกซ์ได้ทำการกำหนดประเภทของผู้ใช้งานไฟล์ออกเป็น 3 กลุ่มใหญ่ ๆ คือ

- owner เจ้าของไฟล์หรือผู้สร้างไฟล์
- group กลุ่มเจ้าของไฟล์
- other or universe ใครก็ได้

โดยสิทธิต่าง ๆ ในไฟล์นั้นจะประกอบไปด้วย การให้สิทธิในการอ่านไฟล์ (read) แก้ไขเปลี่ยนแปลงไฟล์ได้ (write) และสามารถรันไฟล์หรือค้นหาได้ (execute, search) รูปแบบการแสดงสิทธิในไฟล์นั้นจะใช้ตัวเลข 9 บิต หรือเรียกว่า nine permission bit และยังสามารถแทนตัวเลขแสดงสิทธิได้ด้วยเลขฐานแปด ดังตารางในรูปที่ 14



รูปที่ 13 แสดงรูปแบบการ mount ไฟล์



รูปที่ 14

โดยที่

- 3 บิตแรกเป็นการแสดงสิทธิของเจ้าของไฟล์
- 3 บิตที่สองเป็นการแสดงสิทธิของกลุ่มเจ้าของไฟล์
- 3 บิตสุดท้ายเป็นการแสดงสิทธิของผู้ใช้อื่น ๆ

การเปลี่ยนแปลงสิทธิต่าง ๆ ในไฟล์

การเปลี่ยนแปลงสิทธิต่าง ๆ ในไฟล์จะทำได้นั้นมีเพียง 2 กรณีคือ ผู้ที่ทำการเปลี่ยนแปลงนั้นเป็นเจ้าของไฟล์เอง หรืออีกกรณีผู้เปลี่ยนแปลงเป็นผู้ดูแลระบบ (root) การเปลี่ยนแปลงสิทธิต่าง ๆ ของไฟล์ สามารถกระทำได้โดยใช้คำสั่ง chmod ตามด้วยสิทธิที่ต้องการให้เป็น และต่อท้ายด้วยชื่อไฟล์ที่ต้องการเปลี่ยนแปลง ดังรูปที่ 15 ซึ่งเป็นการเปลี่ยนแปลงสิทธิโดยให้ผู้ใช้ทุกกลุ่มนั้นสามารถเอ็กซีคิวต์ไฟล์ได้ นอกจากนี้ยังสามารถใช้คำสั่ง chmod ตามด้วยเลขฐานแปด ซึ่งเป็นค่ากำหนดสิทธิที่เราต้องการ ซึ่งแสดงในตารางรูปที่ 16

```
$ ls -l test
-rwx------   1 snp      operator   10371 Jul 06 10:43 test
$ chmod +x test
$ ls -l test
-rwx--x--x   1 snp      operator   10371 Jul 06 10:43 test
```

รูปที่ 15

Permission encoded by chmod

Octal	Binary	Permission
0	000	none
1	001	--x
2	010	-w-
3	011	-wx
4	100	r--
5	101	r-x
6	110	rw-
7	111	rwx

รูปที่ 16

เช่น คำสั่ง chmod 744 test เป็นการกำหนดให้ไฟล์ชื่อ test นั้น เจ้าของไฟล์สามารถกระทำได้ทุกอย่าง ส่วนกลุ่มของเจ้าของและผู้อื่นสามารถอ่านไฟล์ได้เท่านั้น ไม่สามารถทำการแก้ไขหรือเอ็กซีคิวต์ได้ ดังรูปที่ 17

นอกจากนี้ยังสามารถเปลี่ยนผู้เป็นเจ้าของไฟล์ และกลุ่มเจ้าของไฟล์ได้เช่นกัน โดยใช้คำสั่ง chown (change owner) และ chqrp (change group) ดังรูปที่ 18

```
$ chmod 744 test
$ ls -l test
-rwxr--r--   1 snp      operator   10371 Jul 06 10:43 test
```

รูปที่ 17

```
$ ls -l test
$ chgrp  student test
$ ls -l test
-rwxr--r--   1 snp      student    10371 Jul 06 10:43 test
$ chown tammy test
$ ls -l test
-rwxr--r--   1 tammy    student    10371 Jul 06 10:43 test
```

รูปที่ 18

การกำหนดสิทธิต่าง ๆ ในไฟล์โดยอัตโนมัติ

ในเชลล์จะมีคำสั่ง umask ไว้กำหนดค่าสิทธิต่าง ๆ ในไฟล์โดยอัตโนมัติ เก็บไว้ที่ .profile สำหรับบอร์นเชลล์ และดอร์นเชลล์ หรือ chrc สำหรับซีเชลล์ ชื่อไฟล์ทั้งสองนี้เป็นไฟล์เริ่มต้นในการใช้งานทุกครั้ง การกำหนดเช่นนี้เพื่อที่จะได้ไม่ต้องทำการกำหนดสิทธิหรือเปลี่ยนแปลงสิทธิของไฟล์ทุกครั้งที่มีการสร้างขึ้นใหม่ โดยรูปแบบของคำสั่ง umask จะตามด้วยตัวเลขฐานแปดจำนวน 3 หลัก ซึ่งตัวเลขจะแทนสิทธิต่าง ๆ ในทางตรงข้ามกับการกำหนดสิทธิ์ของไฟล์ ดังตารางรูปที่ 19

โดยทั่วไปแล้ว ค่าของ umask จะถูกกำหนดให้เป็น 022 คือ ให้เจ้าของไฟล์นั้นมีสิทธิเต็มที่ในไฟล์ ๆ นั้น ส่วนผู้ใช้ในกลุ่มและผู้ใช้อื่น ๆ เพียงสามารถอ่านและเอ็กซีคิวต์ไฟล์นั้นได้ แต่ไม่สามารถทำการเปลี่ยนแก้ไขใด ๆ ได้

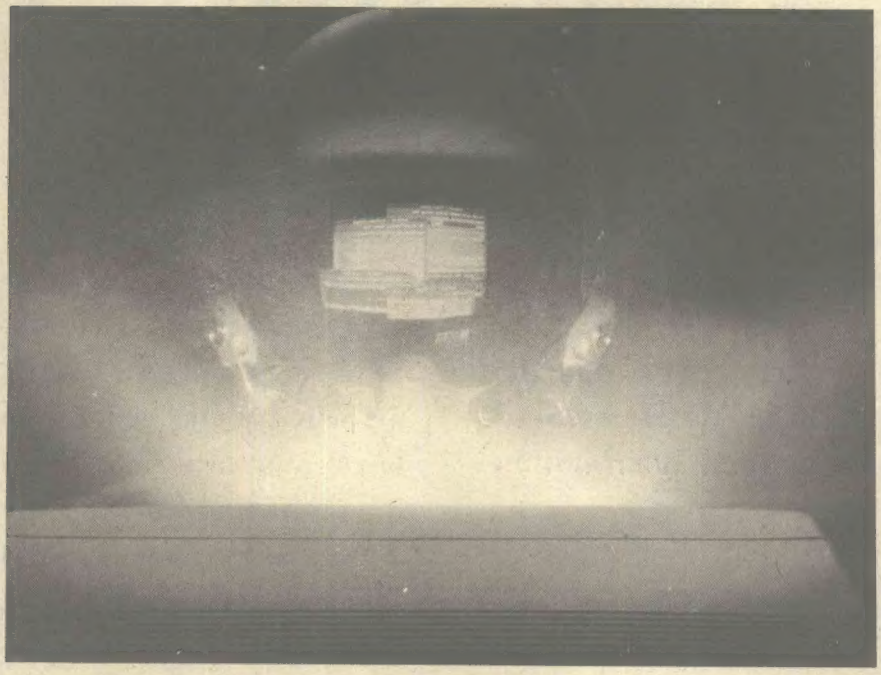
Permission encoded by umask

Octal	Binary	Permission
0	000	rwx
1	001	rw-
2	010	r-x
3	011	r--
4	100	-wx
5	101	-w-
6	110	--x
7	111	none

รูปที่ 19

การกำหนดค่า UID และ GID

การกำหนดค่า UID นั้นเพื่อให้ผู้อื่นที่เข้ามาใช้ไฟล์ (เอ็กซีคิวต์) มีสิทธิเป็นเสมือนกับเจ้าของไฟล์นั้นชั่วคราว คือจะมี EUID เหมือนกับเจ้าของไฟล์ ส่วน UID นั้นยังเป็นของตัวเอง (ปกติเจ้าของไฟล์ (owner) จะแบ่งเป็น real และ effective UID ซึ่งค่าทั้งสองจะเหมือนกัน) แต่ขีดความสามารถในการจัดการไฟล์นั้นยังอยู่ในขอบเขตที่จำกัด ต้องเป็นไปตามกฎเกณฑ์ของโปรแกรมนั้น อย่างเช่น เกมต่าง ๆ จะเก็บคะแนนสูงสุดไว้ ผู้ใช้จะทำการเปลี่ยนแปลงได้ก็ต่อเมื่อเป็นไปตามกฎเกณฑ์ที่โปรแกรมนั้นกำหนด



ส่วนลักษณะการกำหนดค่า GID หรือ setgid นั้นก็เช่นเดียวกัน แต่เป็นในลักษณะของสิทธิของกลุ่มเจ้าของไฟล์แทน คือถ้าผู้ใดก็ตามใช้ไฟล์ (เอ็กซีคิวต์) ที่มีการกำหนดค่า setgid แล้ว ผู้ใช้นั้นจะได้รับสิทธิเสมือนกับเป็นสมาชิกของเจ้าของไฟล์เลยทีเดียว

การกำหนดค่า UID (setuid) และ GID (setgid) นั้นสามารถกระทำได้โดยใช้คำสั่ง chmod 4000 และ 2000 ตามลำดับ อย่างเช่น chmod 4751 test นั้นเป็นการ setuid ให้กับไฟล์ test1 และกำหนดสิทธิไฟล์ให้เจ้าของไฟล์มีสิทธิเต็มที่ สมาชิกหรือกลุ่มเจ้าของไฟล์สามารถอ่านและเอ็กซีคิวต์ได้ ส่วนผู้ใช้อื่น ๆ สามารถเอ็กซีคิวต์ได้อย่างเดียว

รูปแบบการแสดงของไฟล์ที่มีการ setuid หรือ setgid นั้น จะมีการแทนที่ด้วยอักษร s (lower character) ลงบนตำแหน่งของ x (execute bit) โดยหากมีการ setuid และมีการแทนที่ตำแหน่ง x ในชุด 3 บิตแรก หรือชุดสิทธิของเจ้าของไฟล์ หากมีการ setgid นั้นอักษร s จะถูกแทนที่ตำแหน่ง x ในชุด 3 บิตที่สอง หรือชุดสิทธิของกลุ่มเจ้าของไฟล์นั้นเอง และถ้าหากมีการ setuid หรือ setgid แล้วไฟล์นั้นไม่ได้กำหนดให้ทุกคนเอ็กซีคิวต์ได้ ตำแหน่งของอักษร x จะถูกแทนด้วย s (Upper Character) ดังรูปที่ 20

การกำหนด Sticky bit

การกำหนด sticky bit ให้กับไฟล์ใดนั้น เป็นการระบุให้ทำการเก็บไฟล์นั้นไว้ในส่วนของ swap area แม้ในขณะนั้นยังไม่มีการเอ็กซีคิวต์ก็ตาม ซึ่งการที่ไฟล์ใดนั้นอยู่ในส่วนของ swap area แล้วจะทำการเอ็กซีคิวต์ทำได้เร็วขึ้น แต่จะสูญเสียเนื้อที่ swap area บางครั้ง ดังนั้นจึงมีการกำหนด sticky bit ให้กับไฟล์ที่มีการใช้งานบ่อยครั้ง เช่น vi โดยรูปแบบการแสดงสิทธิของไฟล์ที่มีการกำหนด sticky bit จะใช้อักษร t แทน x ในชุด 3 บิตสุดท้าย การกำหนด sticky bit สามารถกระทำได้โดยใช้คำสั่ง chmod โดยที่โหมดของ sticky bit เป็น 1000 ในเลขฐานแปด

```
$ chmod 4751 test
$ ls -al test
-rwsr-x--x   1 snp      operator   10371 Jul 06 10:43 test
```

รูปที่ 20

สรุป

เรื่องราวไฟล์ต่าง ๆ ในยูนิกซ์นั้นยังมีอีกมากมายเลยทีเดียว เรื่องราวข้างต้นนั้นเป็นเพียงเรื่องพื้นฐานของระบบไฟล์ในยูนิกซ์เท่านั้น ซึ่งจะนำเสนอในรายละเอียดของไฟล์เพิ่มเติมในฉบับต่อไป ซึ่งถ้าหากผู้อ่านสนใจจะหาอ่านเพิ่มเติมได้ตามรายชื่อหนังสือในบรรณานุกรม

บรรณานุกรม

1. Andrew S. tanenbaum, operating system, Prentice-Hall Inc., Englewood Cliff New Jersey, 1987

2. Don Libes, Sandy Ressler, Life with UNIX, Prentice-Hall Inc., New Jersey, 1989

3. Evi Nemeth, Garth Snyder, Scott seebass, UNIX SYSTEM ADMINISTRATION HANDBOOK, Prentice Hall, New Jersey, 1989

4. Maurice J. Bach, The Design of the UNIX operating SYSTEM, Prentice Hall., New Jersey, 1986